EUSIPCO 2013 1569746207

SEQUENTIAL SUBSPACE FINDING: A NEW ALGORITHM FOR LEARNING LOW-DIMENSIONAL LINEAR SUBSPACES

Mostafa Sadeghi[a], Mohsen Joneidi[a], Massoud Babaie-Zadeh[a], and Christian Jutten[b]*

[a]Electrical Engineering Department, Sharif University of Technology, Tehran, IRAN.
m.saadeghii@gmail.com, moh872@yahoo.com, mbzadeh@yahoo.com
[b]GIPSA-Lab, Grenoble, and Institut Universitaire de France, France.
christian.jutten@gipsa-lab.grenoble-inp.fr

ABSTRACT

In this paper we propose a new algorithm for learning low-dimensional linear subspaces. Our proposed algorithm performs by sequentially finding some low-dimensional subspaces on which a set of training data lies. Each subspace is found in such a way that the number of signals lying on (or near to) it is maximized. Once we found a subset of the training data that is sufficiently close to a subspace, then we omit these signals from the set of training signals and repeat the procedure for the remaining signals until all training signals are assigned to a subspace. This data reduction procedure results in a significant improvement to the runtime of our algorithm. We then propose a robust version of the algorithm to address the situation in which training signals are contaminated by additive white Gaussian noise. Our simulations on synthetic data and image denoising problem show the applicability and the promising performance of our algorithm.

Index Terms— Linear subspace learning, iterative subspace identification, dictionary learning, sparse residuals

1. INTRODUCTION

In many signal processing applications, e.g. pattern recognition, linear regression, image enhancement, and so on, there are a huge amount of data with very high dimensions. Dealing with these high dimensional data makes the processing tasks very difficult. However, despite of their very high dimensions, it is well-known that many natural signals such as images can be well approximated as a linear combination of a few basis functions [1]. In other words, the *intrinsic dimension* of many natural signals is much less than their length. This fact has led to many dimension reduction algorithms [2].

Consider now a set of L signals, all with dimension n, as the columns of the matrix $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_L]$. Assume that $\mathbf{Y}$ can be partitioned as:

$$\mathbf{Y} = [\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_K], \tag{1}$$

where each $\mathbf{Y}_i$ is a subset of signals that lies on the same low-dimensional subspace. Denoting the basis functions for the ith subspace as the columns of the matrix $\mathbf{D}_i \in \mathbb{R}^{n \times n_i}$, then we have:

$$\mathbf{Y}_i = \mathbf{D}_i \mathbf{X}_i, \tag{2}$$

where $\mathbf{X}_i$ is the coefficient matrix. In the case of many natural signals we have $n_i \ll n$, $\forall i$. In other words, it is assumed that the set of signals lie on a union of low-dimensional subspaces. This simple *model* for the data is very appealing in many signal processing tasks [3].

Principal Component Analysis (PCA) [4] is a well-known tool used for capturing the intrinsic low-dimensional structure of the data. PCA finds a single low-dimensional subspace to represent all training signals, i.e. in the above model it assumes that $K = 1$. As a generalization of PCA, Generalized PCA (GPCA) [5] has been introduced that finds multiple low-dimensional subspaces for a set of signals.

GPCA is a member of a broad class of dimensionally reduction algorithms, known as the *subspace clustering* algorithms [6]. As its name suggests, these algorithms cluster the set of signals into a number of subspaces. K-means [7] is a simple algorithm that clusters the training signals into some ball-shaped (spherical Gaussian) clusters, where all signals of each cluster are represented with a single vector known as the cluster centroid. To deal with the more realistic case in which data clusters are generally hyper-planes, the K-subspace algorithm [8] has been proposed. K-subspace starts with a set of K initial orthonormal matrices, $\mathbf{D}_i$, $i = 1, \ldots, K$, and iteratively assigns each training data to its nearest subspace and then updates each subspace using its own data members by performing Singular Value Decomposition (SVD). This algorithm however, is very sensitive to outliers. Moreover, in very high-dimensional setting it suffers from the high computational load because of performing SVD. Another approach to infer the low-dimensional structures of signals is known as *dictionary learning* in the field of sparse signal processing [9]. In this approach it is assumed that each training signal can be well approximated as a linear combination of a few basis functions. Each basis function is called an *atom* and

*This work has been partially funded by the Iran National Science Foundation (INSF) and by the European project ERC-2012-AdG320684-CHESS.

their collection as the columns of the matrix $\mathbf{D}$ is called *dictionary*. Thus, in the dictionary learning problem we actually have $\mathbf{D} = \cup_i \mathbf{D}_i$. More precisely, a dictionary learning algorithm solves the following minimization problem:

$$\min_{\mathbf{D},\mathbf{X}} \|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_F^2 \quad \text{subject to} \quad \forall i: \ \|\mathbf{x}_i\|_0 \leq T, \quad (3)$$

where $\|.\|_F$ denotes the Frobenius norm, $\|.\|_0$ is the so-called ℓ_0 pseudo-norm that counts the number of nonzero components, and T is a constant. After finding the dictionary, a clustering step is performed using the found coefficient matrix to determine $\mathbf{Y}_i$'s [6]. This approach however, has the drawback that the number of atoms need to be known in advance. Thus one should assume a certain number of atoms, which is not necessarily optimum.

Iterative subspace identification (ISI) [10] is a new approach for subspace clustering. This algorithm is based on the idea that a signal can be represented using a linear combination of the other signals in its own cluster. ISI first selects a signal from the set of training data and finds its sparse representation in terms of the other signals. In this way one subspace is found. The algorithm then proceeds by removing all signals that lie on the found subspace and repeats the same procedure for the remaining signals. This approach is very fast but its accuracy is not acceptable in noisy setting. Like K-subspace, ISI is also very sensitive to outliers, since it tries to represent outliers in terms of a set of non-outlier (inlier) training signals, and thus makes a false cluster.

Up to our best knowledge, many subspace clustering algorithms can be divided into two general categories. The first category iteratively performs a two-stage procedure. The first stage is to assign the training data to their closet subspace. In the second stage, all the found subspaces are updated using their own signals. The algorithms in the second category are based on sequentially finding one subspace on which only a subset of data lies. Once a subspace is found, the signals that are sufficiently close to it are omitted and the same procedure is repeated for the remaining signals. K-subspace belongs to the first category while ISI belongs to the second category.

In this paper we propose to sequentially find a number of linear subspaces for the training data[1]. It is assumed that the dimensions of these subspaces are known in advance. Our algorithm belongs to the second category of subspace clustering algorithms discussed in the previous paragraph. Unlike ISI, our algorithm is robust to noise and especially outliers.

This paper is organized as follows. In Section 2 we describe our algorithm in details. Section 3 presents the results of the simulations.

[1]It is worthwhile mentioning that our algorithm is mathematically similar to [11], which is in the field of robust sensor networks, but these two algorithms are conceptually very different.

2. THE PROPOSED ALGORITHM

We assume, without loss of generality, that the dimensions of the subspaces are all equal to s, i.e. $n_i = s$, $\forall i$. Firstly, we find a subspace $\mathbf{D}_s$ for some of the training data, $\mathbf{Y}$. To this aim, we solve the following general minimization problem:

$$\min_{\mathbf{D}_s,\mathbf{X}} \sum_i \rho(\|\mathbf{y}_i - \mathbf{D}_s\mathbf{x}_i\|_2), \quad (4)$$

where $\rho(.)$ is a loss function. A well-known option for this function is $\rho(x) = x^2$. In this way however, all residuals, i.e. $e_i = \|\mathbf{y}_i - \mathbf{D}_s\mathbf{x}_i\|_2$ $\forall i$, would be relatively small. In other words, the subspace $\mathbf{D}_s$ is fitted averagely to all data. But we want to identify a subspace with the highest population. To this aim, we firstly assume the following model for the data:

$$\forall i: \ \mathbf{y}_i = \mathbf{D}_s\mathbf{x}_i + \mathbf{o}_i, \quad (5)$$

where $\mathbf{o}_i = \mathbf{0}$ if $\mathbf{y}_i \in \text{span}(\mathbf{D}_s)$ and $\mathbf{o}_i \neq \mathbf{0}$, otherwise. In this way, those signals that have a large distance from $\mathbf{D}_s$ are treated as outliers. We consider the following loss function:

$$\rho(x) = \begin{cases} 0 & \text{if } x = 0 \\ 1 & \text{if } x \neq 0 \end{cases}. \quad (6)$$

This function has the desired property that it penalizes equally all signals that do not lie on $\mathbf{D}_s$, i.e. out-of-subspace signals, while it has no loss for the on-the-subspace signals. By substituting this function in (4), we actually obtain a special form of M-estimators [12]. A well-known approach to solve these types of estimation problems is to use the idea of iteratively re-weighted least squares [13]. More precisely, we solve the following iterative minimization problem:

$$\left\{\mathbf{D}_s^{(k+1)}, \mathbf{X}^{(k+1)}\right\} = \underset{\mathbf{D}_s,\mathbf{X}}{\text{argmin}} \sum_i w_i^{(k)} \|\mathbf{y}_i - \mathbf{D}_s\mathbf{x}_i\|_2^2, \quad (7)$$

where[2]

$$\forall i: \ w_i^{(k)} = \frac{1}{\|\mathbf{y}_i - \mathbf{D}_s^{(k)}\mathbf{x}_i^{(k)}\|_2^2}. \quad (8)$$

By starting with $\forall i: \ w_i^{(0)} = 1$, and an initial $\mathbf{D}_s^{(0)}$, each data that has a relatively low distance from $\mathbf{D}_s^{(0)}$ will be more likely to belong to it and in the next iteration will have a larger contribution to the objective function.

2.1. Solving the Minimization Problem

In order to solve (7), we use an alternating-minimization approach, that is, we iteratively minimize (7) over one variable and set the other fixed. The minimization over $\mathbf{X}$ is straightforward and results in the following solution:

$$\mathbf{X}_s^{(k+1)} = \mathbf{D}_s^{(k)\dagger}\mathbf{Y}, \quad (9)$$

[2]To prevent the division by zero, we can add a small positive number to the denominator of $w_i^{(k)}$.

- **Task:** Learning some low-dimensional subspaces from $\mathbf{Y}$.
- **Subspace finding:** Set $r = 1$, $\mathbf{Z} = \mathbf{Y}$ and repeat the following steps until $\mathbf{Z}$ becomes empty:
  a. Set $\mathbf{D}_s = \mathbf{D}_s^{(0)}$ and do the following steps:
    1- $\mathbf{X}_s = \mathbf{D}_s^{\dagger}\mathbf{Z}$
    2- $\mathbf{W} = \text{diag}(w_i = \|\mathbf{z}_i - \mathbf{D}_s\mathbf{x}_i\|_2^2)$
    3- $\mathbf{D}_s = (\mathbf{Z}\mathbf{W}\mathbf{X}_s^T)(\mathbf{X}_s\mathbf{W}\mathbf{X}_s^T)^{-1}$, normalize $\mathbf{D}_s$
    4- If stopping condition is not met go back to step 1.
  b. $w = \{i : \ e_i > \tau\}$
  c. $\mathbf{Z} = \mathbf{Z}(:, w)$
  d. $\mathbf{D}_s^r = \mathbf{D}_s$, $r = r + 1$ and go back to a.
- **Output:** $\mathbf{D}_s^r, \ \ r = 1, 2, \ldots$

Fig. 1. The proposed algorithm. By $\mathbf{Z}(:, w)$ we mean those columns of $\mathbf{Z}$ indexed by w.

where $\dagger$ denotes the Moore-Penrose pseudo inverse, which is defined as $\mathbf{A}^{\dagger} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T$. By defining the diagonal matrix $\mathbf{W}_k = \text{diag}(w_i^{(k)})$, it can be easily verified that the minimization of (7) over $\mathbf{D}_s$ results in the following solution:

$$\mathbf{D}_s^{(k+1)} = (\mathbf{Y}\mathbf{W}_k\mathbf{X}_s^T)(\mathbf{X}_s\mathbf{W}_k\mathbf{X}_s^T)^{-1}, \qquad (10)$$

where $\mathbf{X}_s = \mathbf{X}_s^{(k+1)}$. After a few iterations, we identify the first found subspace, $\mathbf{D}_1$, as $\mathbf{D}_1 = \mathbf{D}_s^{(k+1)}$.

2.2. Sequential Subspace Finding

Once we found $\mathbf{D}_1$, we omit those data that are relatively near to it from the set of training data and repeat the same procedure to find $\mathbf{D}_2$ and similarly the other subspaces. We can use a simple threshold, τ, on the representation errors to decide if a data belongs to a subspace. Another option is to use *the first jump rule* [14]. To explain this method, we define the error vector as $\mathbf{e} = [e_i], \ i = 1, \ldots, L$, and $e_i = \|\mathbf{y}_i - \mathbf{D}_s\mathbf{x}_i\|_2$. We also define $\tilde{\mathbf{e}} = [e_{[1]}, \ldots, e_{[L]}]$ where the entries are sorted increasingly and $e_{[i]}$ denotes the ith largest component of $\mathbf{e}$. This rule then looks for the smallest i such that:

$$e_{[i+1]} - e_{[i]} > \epsilon, \qquad (11)$$

where ϵ is a small positive constant. We then choose $\tau = e_{[i]}$.

Figure 1 summarizes the main steps of our proposed algorithm. The initial subspace, i.e. $\mathbf{D}_s^{(0)}$, can be made by randomly choosing from the set of current training data followed by a normalization. However, as we saw in our simulations, it is better to run the whole algorithm several times and use the subspaces found in each run to initialize the next run.

2.3. Robustness Against Noise

The above algorithm works well only when the data are clean and perfectly satisfy the union of subspaces model. In real applications however, data are usually imperfect. This imperfection may be for example the instrumental inaccuracies, the man-made noise, and the quantization noise. We assume that the imperfection is additive white Gaussian noise (AWGN), and we modify our model to the following form in order to handle the noise:

$$\forall i : \ \mathbf{y}_i = \mathbf{D}_s\mathbf{x}_i + \mathbf{o}_i + \mathbf{n}_i, \qquad (12)$$

where $\mathbf{n}_i \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$. We consider the following loss function for this situation:

$$\rho(x) = \min(x^2, \lambda^2) = \begin{cases} x^2 & \text{if } x \leq \lambda \\ \lambda^2 & \text{if } x > \lambda \end{cases}, \qquad (13)$$

where λ is a threshold to distinguish between the on-the-subspace and the out-of-subspace signals. This parameter is set according to this fact that for the signals on the same subspace, the error vectors i.e. $\mathbf{e}_i = \mathbf{y}_i - \mathbf{D}_s\mathbf{x}_i, \ i = 1, \ldots, L$, are multivariate Gaussian. Thus, for these signals we should have $\rho(x) = x^2$, and for the others $\rho(x) = \lambda^2$, which are treated equally. The value of λ should be large enough to ensure that the real on-the-subspace signals are assigned to one subspace while it should be small enough to prevent the out-of-subspace (outliers) to be included in that subspace. As a rule of thumb, which is suggested in [11], we set λ according to $\lambda = 1.34\sqrt{n}\sigma$.

The function in (13) is non-smooth and thus non-differentiable. We use the following function instead:

$$\rho(x) = \frac{x^2}{x^2 + \lambda^2}. \qquad (14)$$

This function behaves like ℓ_2 norm for small residuals while it ignores large residuals by assigning to them a constant cost. Figure 2 shows some graphs of this function.

We solve the resulting problem in the same way as in (7) and (8) using the following weights:

$$\forall i : \ w_i^{(k)} = \frac{1}{\|\mathbf{y}_i - \mathbf{D}_s^{(k)}\mathbf{x}_i^{(k)}\|_2^2 + \lambda^2}. \qquad (15)$$

With the previous explanations in mind, (8) is a special form of (15), which corresponds to the noise-free case (i.e. $\lambda = 0$).

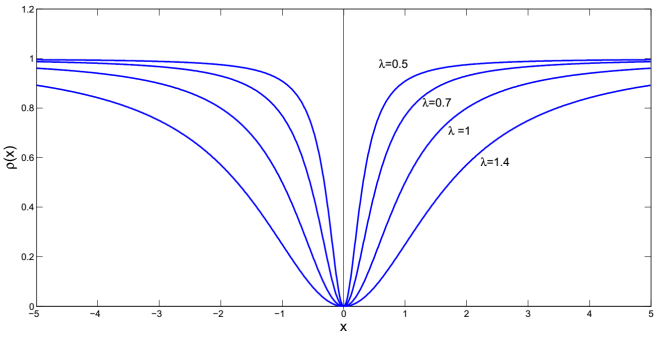

Fig. 2. Graphs of the function in (14) for some values of λ.

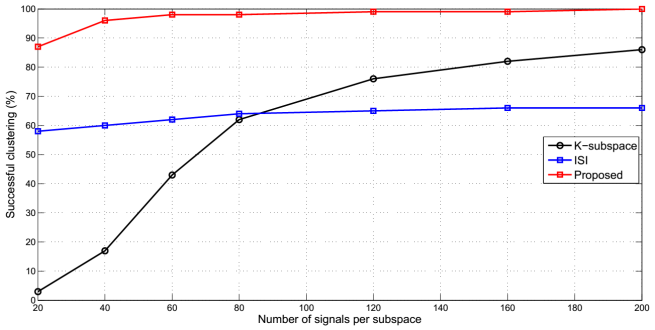

Fig. 3. Percentage of successful clustering versus number of signals in each subspace.

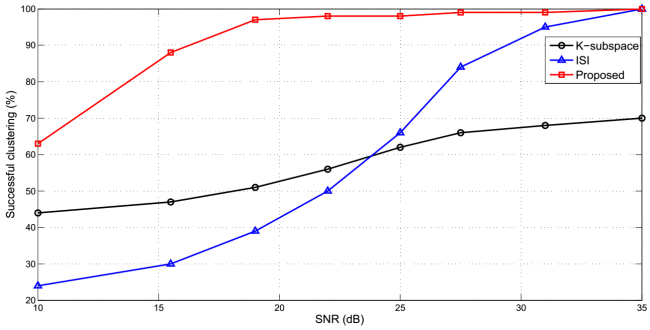

Fig. 4. Percentage of successful clustering versus SNR.

3. EXPERIMENTAL RESULTS

We evaluate the efficiency of our proposed algorithm with two sets of experiments, one on synthetic data and the other on real data. Our simulations were performed in MATLAB R2010b environment on a system with 3.21 GHz CPU and 3 GB RAM, under Microsoft Windows XP operating system. As a rough measure of complexity, we will mention the run times of the algorithms.

3.1. Synthetic Data

In this experiment, we produced 5 disjoint subspaces in $\mathbb{R}^{20}$ all with dimension of 4. The basis for each subspace was generated randomly with independent and identically distributed (i.i.d.) Gaussian zero mean and unit variance entries followed by a normalization. We then produced a certain number of signals all of dimension 20 in each subspace as the training data. To evaluate the robustness of the algorithms against noise, we added AWGN with a certain Signal to Noise Ratio (SNR) to the training signals. We say that a clustering is successful if more than 95% of the signals are correctly assign to their associated subspaces. We repeated each trial 100 times and averaged the results.

Figure 3 shows the percentage of successful clustering versus the number of signals in each subspace where SNR is fixed and equal to 25 dB. As can be seen, K-subspace is very sensitive to the number of signals. In contrary, the number of signals per each subspace has nearly no effect on the successful recovery of ISI and our algorithm. The effect of noise on the successful clustering is plotted in Fig. 4, where the number of signals in each subspace is fixed and equal to 80. This figure shows that ISI is very sensitive to noise while it is not the case for our algorithm. From these two figures we can deduce that the main limitation for ISI is noise while for K-subspace it is the number of signals per each subspace. Also the performance of ISI and our algorithm are similar for relatively high SNR's. The average execution times of K-subspace, ISI, and our algorithm (for Fig. 3) were 0.251, 0.015, and 0.017 seconds, respectively.

Fig. 5. Original, noisy, and denoised images of Barbara by K-SVD and our algorithm.

3.2. Image Denoising

In this experiment we consider the problem of image denoising via sparse approximation [15]. In this approach, an overcomplete dictionary is firstly trained using some blocks of the noisy image as the training data. Then the blocks of the noisy image are denoised over the learned dictionary. We compared our algorithm with K-SVD [15]. Four test images, all of size 256×256, were used in this experiment. For both algorithms we used 15000 blocks of each test image. For our algorithm a number of subspaces with dimension of 5 were learned. Number of atoms was set equal to 256 in both algorithms (for our algorithm the found atoms were firstly sorted according to their variances and then the first 256 sorted atoms were selected). We used Peak Signal to Noise Ratio (PSNR) as the measure of the quality of denoising. Each experiment was repeated 5 times and the averaged results were reported.

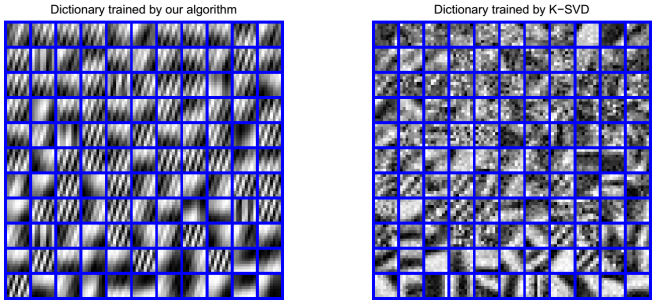

Fig. 6. Some of the atoms learned by our algorithm (left) and K-SVD algorithm (right) for Barbara.

Table 1 summarizes the final results. This table shows the capability of our algorithm in this experiment. Figure 5 shows the original, noisy and denoised images of Barbara by K-SVD and our algorithm. Some of the atoms learned by our algorithm and K-SVD are shown in Fig. 6. As can be seen, the atoms in the dictionary learned by our algorithm successfully recovered the main textures of the original image.

Table 1. Image denoising PSNR results in dB. In each cell two results are reported. Top: results of the K-SVD, and Bottom: results of our proposed algorithm.

σ, PSNR	House	Boat	Lena	Barbara
5, 34.16	39.44	37.41	38.86	38.13
	39.52	37.53	38.91	38.25
10, 28.11	36.08	33.32	35.02	34.10
	36.16	33.48	35.14	34.21
20, 22.11	33.27	29.54	30.26	30.17
	33.46	29.68	30.34	30.28
50, 14.15	28.07	24.81	26.14	25.33
	28.26	24.97	26.49	25.71

4. CONCLUSION

In this paper we proposed an algorithm to learn low-dimensional subspaces embedded in a set of training data. Our algorithm works with sequentially finding a number of low dimensional subspaces for the data. In this way, our algorithm is indeed a robust clustering algorithm that treats those signals that have a relatively large distance from a subspace as outliers. In order to make the algorithm robust in the presence of AWGN, we proposed a robust version of it. Simulation results on both synthetic and real data show the applicability and efficiency of our algorithm. Theoretical analysis of the proposed algorithm is a subject for the future works.

5. REFERENCES

[1] B. A. Olshausen and D. J. Field, "Sparse coding with an overcomplete basis set: a strategy employed by V1?," *Vision Research*, vol. 37, pp. 3311–3325, 1997.

[2] I. K. Fodor, "A survey of dimension reduction techniques," Tech. Rep., LLNL, 2002.

[3] R. G. Baraniuk, V. Cevher, and M. B. Wakin, "Low-dimensional models for dimensionality reduction and signal recovery: A geometric perspective," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 959–971, 2010.

[4] I. T. Jolliffe, *Principal Component Analysis*, Springer, second ed. edition, 2002.

[5] R. Vidal, Y. Ma, and S. Sastry, "Generalized principal component analysis (GPCA)," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 12, pp. 1945–1959, 2005.

[6] E. Elhamifar and R. Vidal, "Sparse subspace clustering: Algorithm, theory, and applications," *IEEE TPAMI*, submitted., Available: http://arxiv.org/abs/1203.1005.

[7] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*, Springer, 1992.

[8] J. Ho, M. Yang, J. Lim, K. Lee, and D. Kriegman, "Clustering appearances of objects under varying illumination conditions," in *Computer Vision and Pattern Recognotion*, 2003, vol. 1, pp. 11–18.

[9] M. Elad, *Sparse and Redundant Representations*, Springer, 2010.

[10] B. V. Gowreesunker and A. H. Tewfik, "Learning sparse representation using iterative subspace identification," *IEEE Trans. on Signal Proc.*, vol. 58, no. 6, pp. 3055–3065, 2010.

[11] V. Kekatos and G. B. Giannakis, "From sparse signals to sparse residuals for robust sensing," *IEEE Trans. on Signal Proc.*, vol. 59, no. 7, pp. 3355–3368, 2011.

[12] P. J. Huber and E. M. Ronchetti, *Robust Statistics*, New York: Wiley, 2009.

[13] R. Chartrand and W. Yin, "Iteratively reweighted algorithms for compressive sensing," in *IEEE ICASSP*, 2008.

[14] Y. Wang and W. Yin, "Sparse signal reconstruction via iterative support detection," *SIAM Journal on Imaging Sciences*, vol. 3, no. 3, pp. 462–491, 2010.

[15] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Trans. on Image Processing*, vol. 15, no. 12, pp. 3736 – 3745, 2006.